AVM FACULDADE INTEGRADA

PÓS-GRADUAÇÃO LATO SENSU
Desenvolvimento de Sistemas com Java

ANDRÉ RESENDE ROCHA

SISTEMA EM JAVA DE CONTROLE DE DATAS EM PRESSUPOSTOS PROCESSUAIS NO STJ

BSB, JANEIRO 2017

AVM FACULDADE INTEGRADA
(PÓS-GRADUAÇÃO LATO SENSU)

ANDRÉ RESENDE ROCHA

SISTEMA EM JAVA DE CONTROLE DE DATAS EM PRESSUPOSTOS PROCESSUAIS NO STJ

Monografia apresentada à AVM Faculdade Integrada como exigência parcial à obtenção do título de Especialista em Desenvolvimento de Sistemas com Java.

Orientador: Wagner Marcelo Sanchez

BSB
2017

Sumário

RESUMO IX

ABSTRACT X

1 INTRODUÇÃO 1

1.1 Tema 1

1.2 Problema 1

1.3 Justificativa 1

1.4 Objetivo geral 2

1.5 Objetivos específicos 2

1.6 Metodologia 2

2. REFERENCIAL TEÓRICO 3

2.1. Engenharia de Software 3

2.2. Linguagem de Programação Java 4

2.3. Banco de Dados 6

3. TECNOLOGIAS EMPREGADAS 7

3.1. Extreme Program 7

3.2. Java 9

3.3. Servlet 10

3.4. Banco de Dados 11

3.5. MySQL 11

4. IMPLEMENTAÇÃO 13

4.1. PREPARAÇÃO DO AMBIENTE DE DESENVOLVIMENTO 13

4.1.1. Instalação do Eclipse 13

4.1.2. Instalação do Maven 14

4.1.2. Resolvendo as dependências do Maven 17

4.2. IMPLEMENTAÇÃO DOS REQUISITOS DO SISTEMA 19

4.2.1. Tela de login 20

4.2.2. Tela de controle de datas 22

5. CONSIDERAÇÕES FINAIS 25

REFERÊNCIAS BIBLIOGRÁFIAS 26

LISTA DE ILUSTRAÇÕES

Figura 1 - Instalação do Eclipse 13

Figura 2 - Escolha do framework 14

Figura 3 – Instalação do Maven 15

Figura 4 – Procurando o Maven 16

Figura 5 – Repositório do Maven 17

Figura 6 – Dependências do Maven 18

Figura 7 – Tela de Login 20

Figura 8 – Código Fonte Tela de Login 21

Figura 9 – Tela de Controle de Datas 22

Figura 10 – Tela de Resposta 23

Figura 11 – Código Fonte Controle de Datas 24

RESUMO

Este projeto tem como intenção desenvolver e apresentar analiticamente as etapas de um sistema de controle de datas automatizado sobre um estudo de caso real. O sistema se baseará na necessidade do Superior Tribunal de Justiça, local em que trabalho, para controlar as datas de pressupostos processuais que são os períodos em que as partes de um processo devem interpor petições. O sistema será composto de telas gráficas desenvolvidas em Java e um banco de dados implementado em MySql, onde serão armazenadas informações dos servidores credenciados à utilizar o sistema. A motivação para este projeto surgiu à partir da demanda dos próprios servidores que fazem a análise dos pressupostos processuais na sessão em que trabalho. Por demandar muito tempo e trabalho na análise não automatizada destas datas, perde-se muito tempo e menos processos são feitos por dia. Com a automatização deste processo a pretensão é que a produtividade aumente já que será mais rápido e fácil a analise dos mencionados pressupostos processuais. Por fim, durante o desenvolvimento do sistema será feita uma explanação sobre os recursos e tecnologias utilizadas no sistema assim como as tecnologias que não foram empregadas, mas que também poderiam fazer parte do projeto.

Palavras-chave: Java, desenvolvimento, MySql, Superior Tribunal de Justiça, pressupostos processuais, automação.

ABSTRACT

This project intends to develop and present analytically the steps of an automated data control system on a real case study. The system will be based on the need of the Superior Court of Justice, where I work, to control the dates of procedural assumptions that are the periods in which the parties to a case must file petitions. The system will be composed of graphical screens developed in Java and a database implemented in MySql, where will be stored information of the servers accredited to use the system. The motivation for this project arose from the demand of the own servers that make the analysis of the procedural presuppositions in the session in which I work. By demanding a lot of time and work in the non-automated analysis of these dates, much time is lost and fewer processes are done per day. With the automation of this process the pretension is that the productivity increases since it will be faster and easier the analysis of the mentioned procedural presuppositions. Finally, during the development of the system an explanation will be made of the resources and technologies used in the system as well as the technologies that were not used, but could also be part of the project.

Keywords: *Java, development, MySql, Superior Court of Justice, procedural assumptions, automation.*

1 INTRODUÇÃO

1.1 Tema

Desenvolvimento de um sistema baseado em Java para controle de datas em análises de pressupostos processuais no Superior Tribunal de Justiça.

1.2 Problema

As partes envolvidas em um processo judicial possuem um prazo específico para a interposição de cada petição e cada recurso. Deve ser feita uma análise, pelos servidores do Superior Tribunal de Justiça, das datas que estes recursos foram interpostos, para que se verifique se o processo será recebido ou não no tribunal. Esta análise atualmente é feita de forma não automatizada o que implica em demora e ineficiência na vazão de processos.

1.3 Justificativa

O tema deste projeto foi escolhido pois será uma grande facilitador nas tarefas diárias do tribunal em que trabalho. A administração pública atual preza pela eficiência em seus processos, o que significa uma melhor utilização dos recursos sem desperdício de recursos. Com este sistema pretendo aumentar a eficiência dos trabalhos dos servidores com isso dando mais vazão aos processos que ingressam na seção em questão. O benefício da eficiência não é o único abrangido por este sistema, já que com um processo automatizado de conferência dos pressupostos processuais tem-se uma maior segurança na veracidade das informações ali aferidas. O sistema será construído na linguagem Java com o auxilio de um banco de dados implementado em MySql. Colocando em prática assim todos os conhecimentos adquiridos no curso de pós-graduação em debate.

1.4 Objetivo geral

Pesquisar e desenvolver explicando detalhadamente os aspectos de um sistema em Java desenvolvido para controlar datas em um sistema de análise de pressupostos processuais no Superior Tribunal de Justiça.

1.5 Objetivos específicos

a) Pesquisar os aspectos de implementação do sistema;
b) Selecionar os recursos e a tecnologia que será utilizada;
c) Sintetizar e implementar o sistema propriamente dito;
d) Explicar os benefícios e limitações das tecnologias escolhidas.

1.6 Metodologia

Inicialmente foi realizada uma pesquisa da necessidades da unidade em que trabalho, procurando melhorar o fluxo de trabalho e a eficiência na rotina da sessão. Foram encontrados alguns pontos de deficiência e estes aspectos foram isolados para uma posterior análise.

A escolha pela implementação deste ponto específico de deficiência na sessão em apreço, se deu pela importância da analise de pressupostos processuais, já que uma falha humana em algum processo provocaria futuramente um julgamento de valor errôneo por parte dos responsáveis ministros.

Com o tema que seria desenvolvido em mente, a próxima etapa foi a escolha da tecnologia que seria utilizada para o desenvolvimento do sistema. Java foi escolhido por se tratar de uma linguagem de desenvolvimento gratuita e multiplataforma.

Os autores aqui referenciados foram os dois mais consagrados sobre o assunto, quais sejam, Paul Deitel e José Augusto Navarro. O Primeiro mais voltado para a área acadêmica traz o conteúdo de forma prolixa e confusa. Já Navarro se preocupa em passar adiante o conteúdo de forma eficiente, ou seja, didaticamente, exemplificando o explicado, no entanto de forma não abrangente.

Este projeto se baseará neste dois autores. Capturando os pontos deficientes em cada e procurando corrigi-los ou ao menos ameniza-los. Fazendo isto, propondo uma abordagem teórica e prática, didática e consistente.

2. REFERENCIAL TEÓRICO

2.1. Engenharia de Software

Atualmente existem diversos tipos de softwares cada um com suas propriedades e particularidades, dentre eles estão os que são destinados à educação e ao controle escolar. Porém o desenvolvimento de um sistema mesmo de pequeno porte requer um alto nível de complexidade, pois muitas são as etapas que precisam ser seguidas, tais como: especificação, desenvolvimento, gerenciamento e evolução. (SOMMERVILLE, 2003).

Estas etapas proporcionam uma maior segurança na construção de um software, mesmo que de pequeno porte. A estatísticas feitas por institutos de pesquisas informam que a grande maioria dos softwares desenvolvidos não atingem a utilidade para o qual foram requeridos. Por esta razão seguir estes passos diminui a probabilidade de que se construam sistemas sem propósitos específicos.

A primeira etapa de especificação talvez seja a mais importante pois irá guiar o restante do trabalho do projeto. Uma especificação errada gerará um sistema errado, ou não requerido. O desenvolvimento é a parte do projeto que tomará mais tempo e esforço. Aqui é realmente construído o sistema com base no que foi especificado anteriormente com o cliente. O gerenciamento é feito quando o software é posto em produção e os usuários reportam sua usabilidade. Conforme os usuários encontram erros e possiblidades de melhorias, o analista de sistema deve ficar atento para que possa aplica-los na próxima etapa, que é a etapa de evolução. A evolução pode ser feita por necessidade ou conveniência. Uma melhoria decorrente de um erro de desenvolvimento ou um melhoria de qualidade na performance ou conteúdo propriamente dito.

Por esta razão ao se desenvolver um sistema de computador deve-se antes de tudo pensar nas especificações, pois possuem a finalidade de descrever as funcionalidades e restrições do software, no desenvolvimento, já que o software deve ser produzido de modo que atenda as suas especificações, na validação, pois o software tem de ser validado para garantir que ele faça o que o cliente deseja, e na possível evolução do mesmo já que ele deve evoluir para atender às necessidades de mudança do cliente. (SOMMERVILLE, 2003).

Mudanças em especificações são inevitáveis. A medida que o software esta sendo construído alterações importantes podem ocorrer nesse meio tempo. Especificações comerciais, governamentais ou mesmo ambientais podem surgir e obrigar mudanças, exigidas pelo cliente, no desenvolvimento do software.

O método XP (eXtreme Programming ou Programação eXtrema) proposto por Kent Beck em 1999 (Beck, 1999) tem como objetivo a excelência no desenvolvimento de software, visando baixo custo, poucos defeitos, alta produtividade, requisitos vagos, e alto retorno de investimento.

Por essa razão, atualmente vem aumentando a aderência a metodologias ágeis. A grande vantagem destas metodologias é a imprecisão dos requisitos iniciais. Como são feitas várias iterações em momentos específicos e em cada iteração é feita uma comunicação e avaliação com o próprio cliente para acompanhar o desenvolvimento do software. Assim as mudanças são feitas imediatamente e os problemas de requisitos falsos, que são descobertos apenas ao final do projeto, são minimizados.

2.2. Linguagem de Programação Java

Em maio de 1995 foi anunciada pela empresa Sun Microsystems a linguagem de programação que faria uma das grandes revoluções no mundo do software. Tal linguagem de programação foi desenvolvida por uma equipe de programadores chefiada por James Gosling e recebeu o nome de Java, que atualmente pertence a Oracle. O fato de a linguagem ser portável para outros sistemas operacionais é o que chamava mais atenção. Embora a linguagem seja constantemente utilizada para o desenvolvimento de aplicações Web ou para dispositivos móveis, ela também é muito usada para aplicações Desktop. (GONÇALVES, 2006).

Atualmente, o grande diferencial da linguagem Java é a portabilidade. A possiblidade de se escrever um mesmo código fonte que será executado em diferentes plataformas e sistemas operacionais, apenas alterando a sua máquina virtual, é uma grande vantagem para o mundo tecnológico heterogéneo em que vivemos. Usaremos neste sistema de controle de datas de analise processual do Superior Tribunal de Justiça a linguagem de programação Java e seus derivados voltados para o ambiente Web.

Um programa Java é formado por várias partes chamadas de classes. Essas por sua vez também são formadas por várias partes chamadas de métodos. Os métodos realizam tarefas e retornam informações ao conclui-las. Dessa forma desenvolvedores podem criar cada parte que precisam para formar um programa Java. Mas na prática, o que geralmente se vê graças à reutilização de classes é o uso das ricas coleções de classes existentes nas bibliotecas de classe Java, também conhecidas como APIs do Java (application programming interfaces). Em Java, existem duas bibliotecas de componentes gráficos que disputam a preferência dos desenvolvedores de sistemas desktop: Swing e SWT (Standard Widget Toolkit). Por essas vantagens é que Java é uma da linguagens de programação que mais cresceu nos últimos anos. (DEITEL, 2005).

Classes podem ser entendidas como abstrações de objetos do mundo real. Antigamente a programação era procedural, ou seja, o programa tinha inicio, meio e fim, e seguia exatamente esta ordem. Com o advento das linguagens orientadas a objeto, a abstração do mundo real para o mundo digital se tornou muito mais natural. Uma classe será uma entidade mapeada do mundo físico, uma casa por exemplo ou um carro. Existem atributos nesta classe, que são as características dela, seus elementos que a compõem, cor, ano, número de rodas, etc. Já os métodos são as funcionalidades desta classe. São as atividades que as classes exercem sobre elas mesmas ou sobre outras classes.

O Java tornou-se a linguagem preferida para implementar aplicativos baseados na internet e software para dispositivos que se comunicam em uma rede. Ele não e mais utilizado simplesmente para tornar as paginas da World Wide Web mais dinâmicas tornou-se a linguagem preferida para atender as necessidades de programação corporativa de muitas empresas.[Deitel 2005]

Várias são as formas de se implementar Java na internet. Servlets, JSP, Applets, JSF entre outras. Esses métodos acompanham todas as funcionalidades já mencionadas e descritas anteriormente inerentes à linguagem Java. A metodologia escolhida para este

projeto específico foi a Servlet. Poderosa ferramenta para implementar aplicativos em Java baseados na Internet. Posteriormente iremos integrar este aplicativo a um banco de dados para persistir os dados.

2.3. Banco de Dados

Ao se desenvolver um software deve-se pensar na função de armazenagem de dados, pois é ela quem gerencia como eles serão persistidos e como serão tratados pelos programas que estão sendo executados no sistema. A essa função dá-se o nome banco de dados, que basicamente é um arquivo ou conjunto de arquivos que guardam algum tipo de informação e que estão relacionadas entre si de alguma maneira, como por exemplo, por meio de campos comuns para que a mesma possa ser acessada ou alterada. (DENNIS & WIXOM, 2005).

A persistência dos dados é essencial atualmente. A informação se tornou o maior ativo de qualquer organização e o diferencial competitivo das empresas. As informações são valiosas e ditam os caminhos futuros das organizações. Os bancos de dados exercem um papel muito importante na retenção destes dados das empresas que de outra forma teriam que armazenar e analisar os dados e informações de forma manual.

Sob uma perspectiva de desenvolvimento o banco de dados relacional é muito mais fácil de trabalhar já que esta baseado em coleções de tabelas, e cada tabela possui um campo ou campos que representam a chave primária, cujo valor é diferente para cada linha da tabela. Desta forma as tabelas são relacionas entre si, colocando a chave primária de uma como uma chave estrangeira na tabela relacionada. Grande parte dos sistemas de gerenciamento de bancos de dados relacionais utilizam uma técnica conhecida como integridade referencial, pois esta assegura que os valores que unem as tabelas por meio de chaves primarias e estrangeiras são válidos e estão sincronizados. (DATE, 2000).

Vários são os tipos de banco de dados no mercado. Existem os bancos de dados hierárquicos, orientados a objetos e o relacional, entre outros. O modelo relacional é o que mais se aproxima a antiga forma de armazenar dados em ficheiros. Tabelas são criadas e divididas em atributos ou campos daquela entidade. São feitos relacionamentos entre estas tabelas para melhor recuperação dos dados. Um gerenciador de banco de dados faz a função de gerente e seguranças das informações,

provendo um ambiente seguro e estável para a manipulação de informações em um ambiente informatizado. O gerenciador de banco de dados escolhido para este projeto foi o MySQL por ser gratuito, portável e suficientemente robusto.

Niederauer (2005) define MySQL com um SGBD – Sistema de Gerenciamento de Banco de Dados – relacional que utiliza a linguagem padrão SQL, e é largamente utilizada em aplicações para a internet, sendo a mais popular entre os bancos de dados com código fonte aberto.

3. TECNOLOGIAS EMPREGADAS

3.1. Extreme Program

Em meados dos anos de 2001 um grupo de pesquisadores, insatisfeitos com os fracassos de projetos de softwares e com a metodologia empregada para tal, se reuniram para discutir mudanças e implementar melhorias na construção de software.

Após algum tempo de pesquisa, esses especialistas publicaram um manifesto, que ficou conhecido como *Manifesto for Agile Software Development* [FOWLER 01]. Esse manifesto destacava quatro valores principais:

a) Indivíduos e iterações mais que processos e ferramentas;
b) Software funcional mais que documentação detalhada;
c) Colaboração do Cliente mais que negociação de contratos;
d) Responder às mudanças mais que seguir um plano.

Primeiramente percebeu-se que as metodologias de construção de software tradicionais se empenhavam muito em aplicar as melhores tecnologias e processos e não davam a devida atenção ao cliente ou ao usuário do sistema. Sendo este de papel de papel fundamental na especificação do software. As metodologias ágeis pregam que o cliente deve participar ativamente da construção do sistema em todas as suas etapas e não apenas no começo.

As extensas documentações elaboradas pelas metodologias tradicionais consumiam muito tempo de mão de obra especializada e por vezes era apenas um

documento formal do sistema. O Extreme Program não proíbe a produção de documentação mas recomenda que a documentação seja enxuta e que o código será a própria documentação do sistema em si.

Novamente voltado para o cliente as metodologias ágeis prezam pela confiança e a colaboração mutua ao invés de se prender em contratos rígidos e imutáveis. As mudanças nas especificações de software são inevitáveis por isso prezar pela iteração e colaboração constante do cliente em todos os momentos do projeto é essencial.

A Extreme Programming (XP) é uma metodologia ágil para equipes pequenas e médias que desenvolvem software baseado em requisitos vagos e que se modificam rapidamente (BECK, 2004). Esta metodologia esta baseada em quatro valores principais, quais sejam:

a) Comunicação;
b) Simplicidade;
c) Feedback;
d) Coragem.

A comunicação deve-se fazer presente entre o cliente e a equipe técnica, entre os stakeholders e entre a própria equipe de desenvolvimento. A simplicidade se reflete no código, já que as funcionalidade mais importantes, essenciais, devem ser entregues primeiro. O mais importante é a satisfação do cliente e o quanto antes entregar algo executável a ele. O feedback é muito importante pois como já foi mencionado os requisitos mudam a todo momento e saber quando um requisito foi alterado o quando antes é menos prejudicial ao projeto. E por último os integrantes da equipe de desenvolvimento devem ter coragem para experimentar, atrever-se e se arriscarem para que os resultados do projeto possam ter valor para o cliente.

As vantagens de se aplicar uma metodologia ágil em contraposição a uma tradicional são comprovadamente eficazes. A taxa de falha em projetos é menor. O atingimento da meta e da expectativa do cliente em relação ao produto final são estatisticamente maiores do que os produzidos sob metodologias tradicionais. Atualmente é uma tendência e uma melhoria na engenharia de software sem precedentes.

3.2. Java

Java é uma linguagem de programação orientada a objetos. A orientação a objetos é uma forma de pensar o código fonte, onde, as entidades do mundo real serão mapeadas em classes com comportamentos e qualidades próprias. Estes comportamentos e qualidades serão transformados em métodos e atributos dentro de uma classe Java. Esse processo de análise consiste em identificar quais objetos existem dentro do universo que se pretende automatizar e quais são os atributos e ações desses objetos (CORREIA, 2006).

Uma classe pode ser entendido como uma “coisa” física, por exemplo, um carro, uma casa ou uma moto. Pode-se definir classe como uma equação ou uma conta bancária, neste caso puramente mental, pois não existiria uma “coisa” física que possa impressionar nossos sentidos para que estes o percebam como um objeto físico. (CORREIA, 2006)

Alguns conceitos importantes são usados na orientação a objetos para melhor relacionar o mundo real ao mundo computacional. Entre outros temos a hierarquia de classes, o encapsulamento e o polimorfismo.

O encapsulamento pode ser utilizado para diminuir o trabalho ao desenvolver um novo sistema, sendo esta a maior vantagem (COAD, 1991). Encapsular significa esconder detalhes de implementação da classe, não permitir que outras classes ou objetos violem propriedades ou métodos da classe encapsulada. Basicamente transformando os métodos de acesso a classe com visibilidade pública e os atributos ou qualidades desta classe privados, onde apenas a própria classe tenha acesso. A vantagem do encapsulamento é que este disponibiliza o objeto com toda a sua funcionalidade sem necessidade de saber qual o seu funcionamento interno e como armazena os dados que são recuperados (CORREIA, 2006).

A hierarquia de classes pode ser sintetizada pela funcionalidade da orientação a objetos chamada de Herança. Herdar algo de uma classe faz com que o que foi herdado não precise ser reescrito, tornando assim o trabalho dos analistas e programadores mais eficiente e econômico. Para exemplificar a herança pode-se utilizar a genética (CORREIA, 2006). O filho herda características genéticas dos pais e, por sua vez, repassa estas características aos seus filhos. São identificados dois tipos de herança, simples e múltipla. Denomina-se herança simples quando uma classe herda características de

apenas uma superclasse (CORREIA, 2006). Herança múltipla é quando uma classe herda características de duas ou mais superclasses (CORREIA, 2006). Java não possuía a possibilidade de se implementar a herança múltipla. Mas esta funcionalidade pode ser mascarada com o uso de Interfaces.

3.3. Servlet

Os Servlets são a base do desenvolvimento de aplicações web utilizando a linguagem Java e uma das suas mais importantes tecnologias. Em uma aplicação web Java, os Servlets são responsáveis por receberem requisições HTTP de páginas web, fazer os processamentos no servidor e devolver os resultados da requisição. Kurniawan (2002)

Entre outras tantas tecnologias de desenvolvimento Web, os Servlets agregam algumas vantagens que fazem com que esta linguagem seja ideal para a implementação deste sistema em questão, qual seja, o controle de datas em sistemas de analise processual do Superior Tribunal de Justiça.

O desempenho dos Servlets é maior do que de outras tecnologias. Isto é possível pois ele não precisa criar um novo processo para cada requisição do cliente. Todas as requisições são mantidas por um único processo que é gerenciado pelo web container. Tornando assim o processo de requisição mais eficiente e rápido.

A portabilidade nativa do Java é estendida aos Servlets. Uma vez criado o código do aplicativo este não precisa ser modificado para cada sistema operacional especifico no qual se deseja rodar o sistema. A portabilidade talvez seja uma das grandes vantagens das tecnologias Java.

Aplicações robustas. Esta é uma outra grande vantagens dos Servlets. A possibilidade de usar todas as camadas de proteção e gerenciamento de uma máquina virtual Java, proporciona a esta tecnologia de desenvolvimento Web um ambiente seguro e confiável para suas aplicações.

Um Servlet é um pequeno programa que é executado em um servidor Web. Eles recebem e respondem as aplicações geralmente através de HTTP. Ele consegue gerar páginas dinâmicas para as camadas de apresentação. Sem o contêiner Web os Servlets não seriam possíveis. Um bom e popular contêiner é o Tomcat o qual será usado neste projeto.

3.4. Banco de Dados

Um banco de dados é uma coleção de dados persistentes, usada pelos sistemas de aplicação de uma determinada empresa. (DATE 2004)

Persistir os dados é mantê-los seguros em algum repositório não volátil, podem assim que estas informações sejam recuperadas para posterior análise ou uso. Os bancos de dados relacionais usam tabelas para armazenar os dados. Cada tabela é uma entidade do mundo real, por exemplo, alunos de uma faculdade seria uma tabela em um banco de dados relacional. Cada tabela possui atributos ou campos que são as qualidades desta entidade. No caso do aluno seus campos na tabela poderiam incluir: nome, cpf, matricula, turma, etc.

Em outras palavras, um banco de dados é um local onde são armazenados dados necessários à manutenção das atividades de determinada organização, sendo este repositório a fonte de dados para as aplicações atuais e as que vierem a existir. (ELMASRI e NAVATHE 2011)

As vantagem da utilização de um banco de dados são inúmeras, entre elas podemos destacar as seguintes: controle centralizado dos dados, já que os dados ficam armazenas apenas em um único local facilitando o gerenciamento e recuperação dos dados; Controle de redundância pois dados duplicados não são armazenados comprimindo o espaço utilizado pelos bancos de dados e evitando erros por duplicação de informações. Gerenciamento facilitados. Este gerenciamento é feito pelos gerenciadores de bancos de dados automatizados. No mercado existem vários e neste projeto específico iremos utilizar o MySQL.

Um sistema de banco de dados é “um sistema computadorizado cuja finalidade geral é armazenar informações e permitir que os usuários busquem e atualizem essas informações quando as solicitar. (DATE 2004)

3.5. MySQL

Um SGBD (Sistema Gerenciador de Banco de Dados) é uma coleção de dados inter-relacionados e também de programas utilitários que são utilizados para acesso e manipulação a estes dados. A principal função do SGBD é proporcionar um ambiente

conveniente e eficiente para recuperação e armazenamento das informações. (Silberschatz e Korth 2006)

O MySQL é um gerenciador de banco de dados com licença livre e código aberto, portanto gratuita, além disso a razão de ter escolhido o MySQL para aplica-lo neste projeto específico é a facilidade de utilização e a robustez do sistema. Quanto a sua distribuição, o MySQL reúne diversos programas e bibliotecas de cliente (*client*), ferramentas administrativas e API'S e muitos outros programas desenvolvidos e disponibilizados por colaboradores e parceiros (LIMA, 2003). Isso facilitara e automatizara a maioria do trabalho do banco de dados, sendo que o analista pode-se focar na programação do sistema em si, não se preocupando com detalhes de implementação do banco de dados.

O gerenciador de banco de dados MySQL é um banco de dados relacional, desenvolvido para qualquer plataforma. Ou seja ele poderá ser utilizado em Linux, Windows e IOS. Ele é um servidor multiusuário, multitarefa e compatível com o padrão SQL. SQL é a linguagem utilizada para comunicação do gerenciador com o banco propriamente dito. Esta linguagem ira servir para fazer as operações em cima dos dados no banco de dados. Entre outras podemos destacar: select, insert, delete, remove, alter table, etc.

O MySQL tem capacidade de manipular até 50 milhões de registros. O que já é extremamente razoável para aplicações simples e grandes aplicações comerciais. Ele é escrito em C e C++ e permite conexões via ODBC, que é a via de comunicação entre a linguagem Java que iremos utilizar, no caso o Servlet, e o banco de dados.

Neste projeto usaremos o MySQL para fazer a persistência dos dados do usuário no banco de dados. Quando usuário se cadastrada no banco de dados fica mais fácil fazer a auditoria e o acompanhamento das ações deste usuário no sistema. Questões de segurança como: quem acessou o sistema, a que horas acessou, com que permissão ou senha, etc. Estas e outras questões de segurança são muito importantes em momentos de auditoria dentro da empresa.

O MySQL foi escolhido por ser um banco de dados gratuito e simples de usar. No entanto é um banco de dados robusto que oferece todas as ferramentas de bancos de dados proprietários que serão necessárias em nosso sistema. Com a facilidade de ter uma interface gráfica que pode ser acessada via browser.

4. IMPLEMENTAÇÃO

4.1. PREPARAÇÃO DO AMBIENTE DE DESENVOLVIMENTO

4.1.1. Instalação do Eclipse

Neste projeto iremos utilizar a ferramenta de desenvolvimento Eclipse para construir nosso sistema em Java. Eclipse é um IDE para desenvolvimento Java que fornece todo o suporte a plug-ins necessários ao nosso projeto. O Eclipse é uma ferramenta open source iniciado pela IBM.

Para utilizar o Eclipse primeiramente tivemos que instalar o JDK, que é um conjunto de ferramenta de desenvolvimento que inclui o compilador Java. Posteriormente foi baixado o Eclipse do próprio site http://eclipse.org. A versão utilizada é o Eclipse Juno4.2.

Para fazer a instalação do Eclipse pelo site oficial entre na sessão de downloads e escolha a opção Eclipse IDE for Java EE Developers associado ao seu sistema operacional apropriado. No nosso caso foi instalada a versão de 64 bits.

Figura 1 - Instalação do Eclipse

Após fazer o download a tela de instalação irá lhe guiar para concluir a instalação do Eclipse em sua máquina.

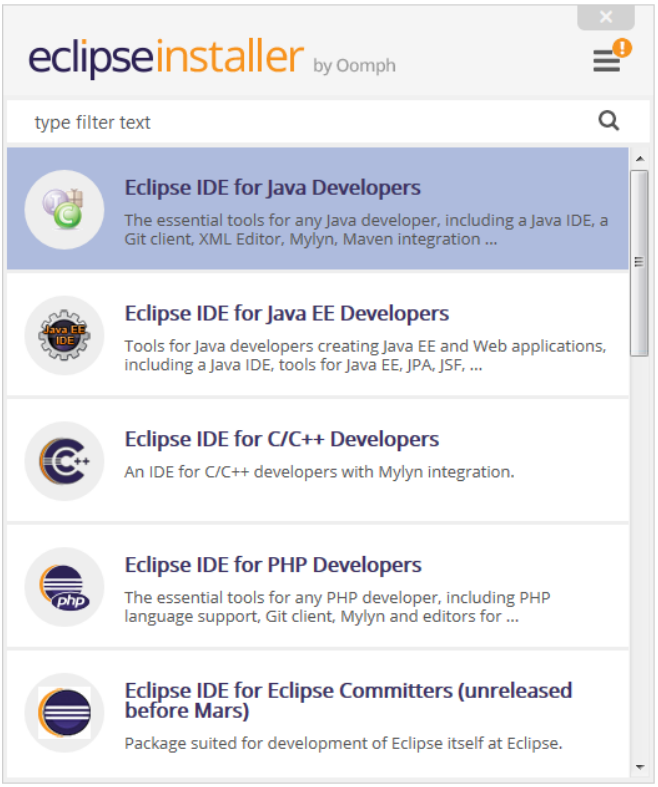

Figura 2 - Escolha do framework

4.1.2. Instalação do Maven

O Maven é uma ferramenta de gerenciamento, construção e implantação de projetos que te ajuda no processo de gerenciamento de dependências, incluindo automaticamente as dependências dos arquivos .jar de seu projeto java, e no de build, geração de relatórios e de documentação.

Para usar o Maven, você precisa ter o JDK instalado. Você também pode integrá-lo à sua IDE. No caso vamos integra-lo em nossa IDE Eclipse, instalando o plug-in do Maven diretamente no Eclipse.

Entre as atribuições do Maven estão inclusas:

a) Facilitar a compilação do código, o empacotamento (JAR, WAR, EAR, ...), a execução de testes unitários, etc.

b) Unificar e automatizar o processo de geração do sistema. Nada mais de uma coleção de passos e scripts a serem executados manualmente.
c) Centralizar informações organizadas do projeto, incluindo suas dependências, resultados de testes, documentação, etc.
d) Reforçar boas práticas de desenvolvimento, tais como: separação de classes de teste das classes do sistema, uso de convenções de nomes e diretórios, etc.
e) Ajuda no controle das versões geradas (releases) dos seus projetos.

Para instalar o Maven no Eclipse, clique em Help no menu do Eclipse e selecione Install New Software. Será aberta a tela de pesquisa de componentes que podem ser instalados no Eclipse. Este componentes são acessórios plug-ins que adicionam funcionalidades que não estão instaladas por padrão na instalação default.

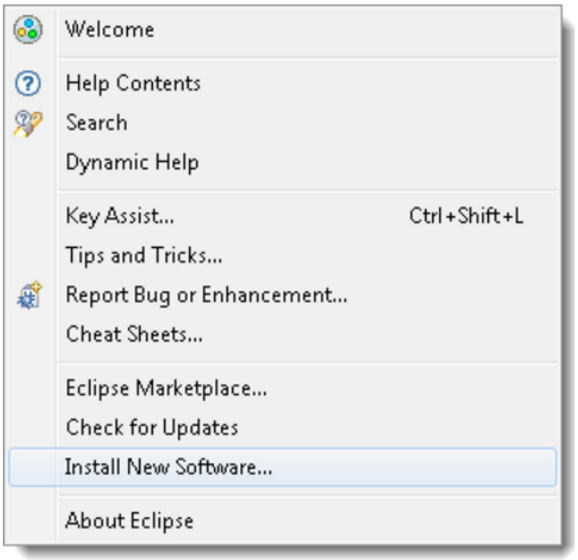

Figura 3 – Instalação do Maven

Ao clicar em “Install New Software” irá abrir uma tela para pesquisar o componente que deseja ser instalado. Como mostra a figura a seguir. Para instalar o Maven basta digitar “Maven” no campo de pesquisa e seleciona-lo na janela logo abaixo que irá mostrar os componentes encontrados. Selecione o Maven e clique em OK. Aceite os termos de uso e depois da instalar ser concluída será necessário reiniciar o Eclipse para que as atualizações possam ser concretizadas.

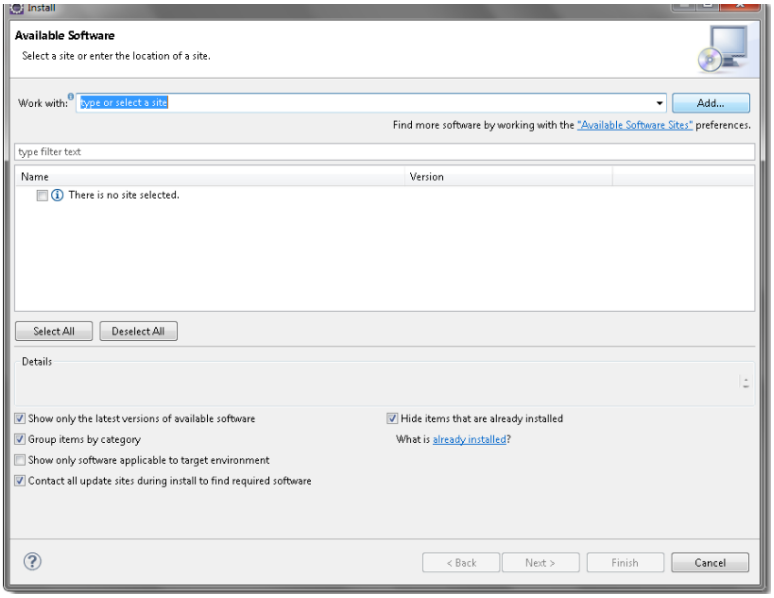

Figura 4 – Procurando o Maven

O Eclipse fornece uma interface de instalação de software como a apresentada a cima, mas também oferece, nas versões mais novas uma outra interface mais dinâmica e didática. É o mercado do Eclipse. Esta funcionalidade pode ser acessada no menu do Eclipse e ao acessa-la o usuário pode escolher de forma muito mais intuitiva as ferramentas ou plug-ins que deseja incluir no Eclipse.

Embora o mercado do Eclipse seja intuitivo e fácil de usar, instalando o software pelo menu de instalação de aplicativos o usuário tem opções mais avançadas de configuração da instalação. Cada componente aqui pode ser escolhido separadamente para ser instalado no Eclipse. Dando ao usuário maior liberdade de escolha sobre os componente que irá realmente precisar sem ter que instalar funcionalidades desnecessárias deixando o programa sobrecarregado.

Na tela que se abre digite Maven no campo Name e em Location digite o seguinte endereço:

http://download.eclipse.org/technology/m2e/releases/

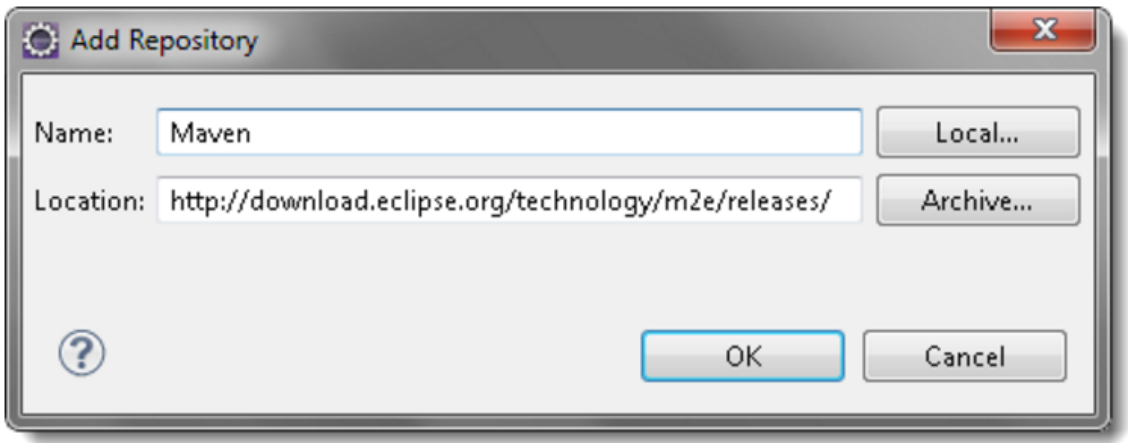

Figura 5 – Repositório do Maven

Feito isto clique em Next e aceite os termos de uso do Maven. O Eclipse irá instalar o Maven e quando concluir será necessário reiniciar o Eclipse para que as alterações possam ser concluídas. Quando o Eclipse reiniciar o Maven estará instalado e pronto para ser usado.

4.1.2. Resolvendo as dependências do Maven

Para a realização deste projeto iremos precisar de algumas bibliotecas especificas que serão importadas ao nosso projeto com a ajuda do Maven. A primeira biblioteca que iremos importar é a do JSF e do PrimeFaces. PrimeFaces é uma implementação do JSF que permite criar componentes visuais pré-codificados. No site do PrimeFaces há uma sessão de demonstração para que o programador veja os componentes disponíveis desta biblioteca e copie para seu projeto fazendo as devidas alterações necessárias.

Para incluir as bibliotecas do JSF e do PrimeFaces em nosso projeto precisaremos abrir o arquivo pom.xml. Este arquivo contém as configurações do nosso arquivo Maven. Neste arquivo é possível incluir as dependências de forma manual, por inclusão de código dentro do arquivo xml ou então de forma gráfica com o auxilio do Eclipse.

Iremos incluir as dependências pela forma gráfica já que é mais fácil e o próprio Maven já verifica a existência de inconsistência entre dependências. Para isso abra o arquivo pom.xml e na parte inferior da janela clique na aba “Dependencies”. A janela abaixo será mostrada:

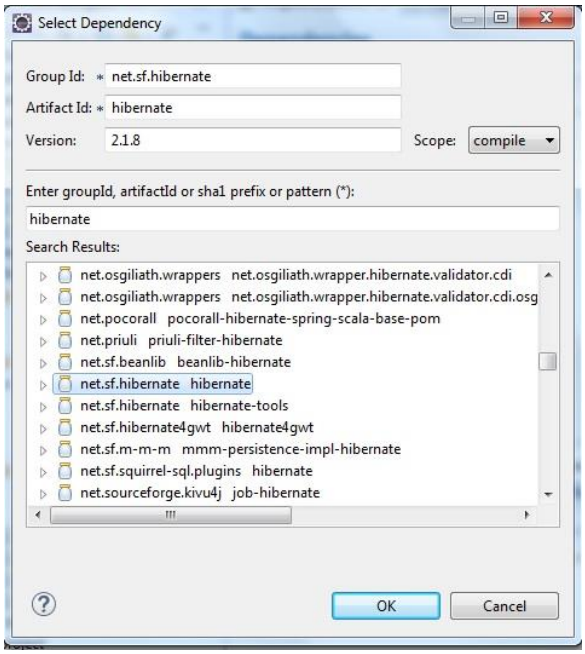

Figura 6 – Dependências do Maven

Para adicionar uma dependência basta preencher os dados da dependência como: “group id”, “artifact id” e “version” e clicar em OK. Ou então pesquisar por uma dependência digitando alguma referencia na barra de pesquisa.

Portanto para instalar o JSF e o Prime faces basta procurar estas dependências na barra de pesquisa selecionar o componente que deseja instalar e clicar em OK. Automaticamente o Maven instalará estas dependências em seu projeto.

Por último precisamos incluir as bibliotecas da JPA e do Hibernate que iremos utilizar em nosso projeto. O Hibernate será usado para fazer a persistência dos dados do usuário do sistema. Basicamente será criada uma classe chamada “Usuário" contendo as informações de Login e Senha.

A Java Persistence API (JPA) é a especificação padrão para o gerenciamento de persistência e mapeamento objeto relacional, surgida na plataforma Java EE 5.0, e que foi baseado no Hibernate proposto pelo projeto Via Digital. A JPA tem o intuito de simplificar o desenvolvimento de aplicações Java que utilizam persistência de dados e possui uma completa especificação para realizar mapeamento objeto relacional, utilizando anotações da linguagem Java. Também dá suporte a uma rica linguagem de consulta, semelhante à SQL, permitindo consultas estáticas ou dinâmicas. Com isso, adotaremos a JPA que atende nossos requisitos e vai facilitar muito a persistência dos dados no nosso banco de dados.

Para incluir as bibliotecas da JPA e Hibernate em nosso projeto precisaremos abrir o arquivo novamente o arquivo pom.xml e fazer os mesmos passos que fizemos para incluir as dependências do JSF e do PrimeFaces. Basta agora pesquisar o artefato Hibernate e clicar em OK para instalar as dependências.

4.2. IMPLEMENTAÇÃO DOS REQUISITOS DO SISTEMA

O sistema de controle de datas que propusemos implementar neste projeto, basicamente apresenta ao usuário uma tela de login. A tela de login pedira as informações de nome de usuário e senha cadastrados no sistema.

Feita a autenticação no sistema a próxima tela que será apresentada ao usuário é a tela principal de nosso sistema. Nela irão aparecer os campos de Data Inicio, que é a data da ultima denegação do recurso, e data fim que é a data em que o recurso no STJ foi interposto e a qual será analisada, estas datas não poderão ultrapassar o limite de 15 dias para que o recurso tenha efeito.

Os processos oriundos de outras instancias poderá ser impetrado recursos no STJ. Estes recursos estão disponíveis ao advogado e ás partes para interposição quando os outros tribunais onde os processos são originados negaram o recurso das partes. Se o tema tratado no processo ferir alguma norma infraconstitucional federal estes recursos, poderão ser interpostos pela parte. No entanto é necessário que ele seja interposto no prazo de 15 dias após a ultima decisão denegatória.

O sistema então terá que calcular se os 15 dias foram respeitados apresentando uma mensagem positiva na tela ou caso contrario, se o recurso interposto ultrapassou os 15 dias uma mensagem de que o recurso não é tempestivo, sendo assim, os servidores do STJ podem dar o tratamento adequado àqueles recursos que não poderão prosseguir o fluxo normal dentro do tribunal.

4.2.1. Tela de login

A tela de login será implementada conforme a figura abaixo:

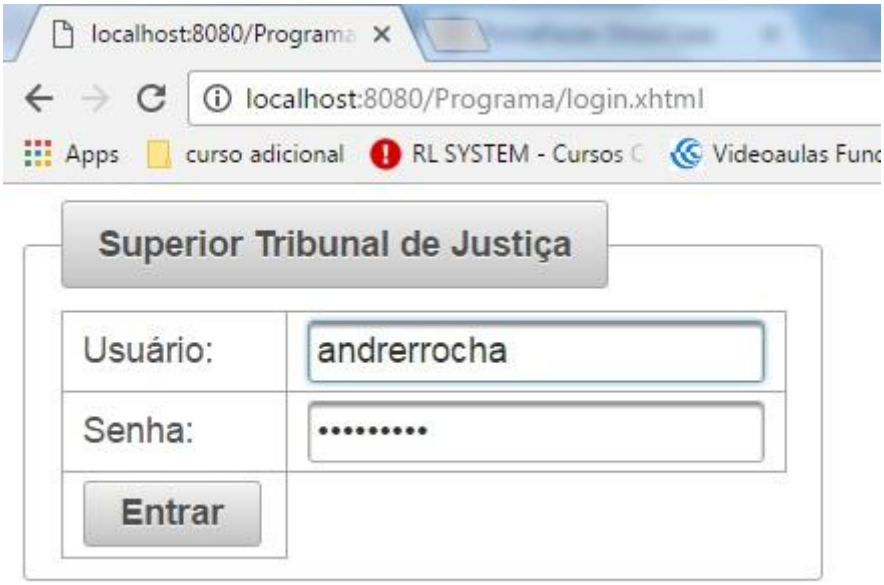

Figura 7 – Tela de Login

Apresentaremos uma mensagem simples de "Superior Tribunal de Justiça" seguido dos campos "Usuário" e "Senha". Após preencher os dados corretamente e clicar no botão Login o usuário estará registrado no sistema e poderá utilizar as funções de controles de datas de processos recursais.

A implementação da tela de login foi feita em JSF. O framework JavaServer Faces(JSF), o qual atua principalmente nas camadas de Visão e Controle do modelo MVC, possui como principais características facilitar o desenvolvimento de componentes personalizados e adotar um modelo de programação orientada a eventos, tentando se assemelhar ao desenvolvimento desktop.

Uma outra característica do JSF é a utilização apenas de requisições síncronas, ou seja, a cada evento invocado pelo cliente é criada uma requisição que segue até o servidor para realizar algum processamento. Em seguida uma nova página é totalmente construída, mesmo que esta seja igual à anterior, e exibida no browser do cliente, que por sua vez espera por todas essas ações sem poder interagir com o sistema.

O código fonte da tela de login é apresentado a seguir:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:p="http://primefaces.org/ui">
<h:head>
</h:head>
<h:body>
  <h:form>
    <p:fieldset legend="Superior Tribunal de Justiça"
style="width:300px">
      <p:panelGrid columns="2">
        <h:outputLabel value="Usuário: "></h:outputLabel>
          <p:inputText value="#{usuario.usuario_nome} "/>
          <h:outputLabel value="Senha: "></h:outputLabel>
          <p:password value="#{usuario.usuario_pass}"id="pass"
feedback="true"/>
          <p:commandButton value="Entrar"></p:commandButton>
       </p:panelGrid>
    </p:fieldset>
</h:form>
</h:body>
</html>
```

Figura 8 – Código Fonte Tela de Login

Um arquivo JSF inicia com a declaração normal de um arquivo xml, informando sua versão que sempre é a “1.0” e a codificação do texto que no caso brasileiro é a UTF-8 que são os caracteres latinos.

Posteriormente é preciso declarar as bibliotecas de vocabulário para serem utilizadas em nossa aplicação. São como API’s de componentes que iremos utilizar. Esta declaração é feita com a tag: “xmlns”.

A partir deste ponto a programação segue a estrutura de um documento html onde precisamos declarar as tags “<head>”, “<body>” e dentro desta toda a lógica de nossa aplicação.

O comando “<form>” cria uma área de formulário onde poderemos incluir campos de texto, entrada de dados e botões. Todos os componentes que estiverem dentro desta tag serão tratados de forma conjunta.

Por fim dentro deste formulário criamos um fildset para acomodar nossos componente, um panelGrid que faz o layout dos componente e as labels os inputTexts e o botão.

4.2.2. Tela de controle de datas

Feito o login no sistema ao usuário será apresentada a tela de Controle de Datas. É a tela principal do sistema que permitira que o usuário insira uma data em um campo denominado “data inical” e outra data no campo denominado “data final”. O sistema irá calcular, ou seja, fazer a diferença entre essas duas datas e informar se o processo é tempestivo ou não. Para ser tempestivo as datas não podem ser distantes mais do que 15 dias. Caso contrário o processo não poderá seguir seu curso normal dentro do tribunal e precisará ser enviado para outra sessão. Além dos campos de datas será disponibilizado um botão para que o usuário submeta a pesquisa ao sistema.

A tela de Controle de Datas também é feita em JSF assim como a tela de Login e é apresentada logo abaixo.

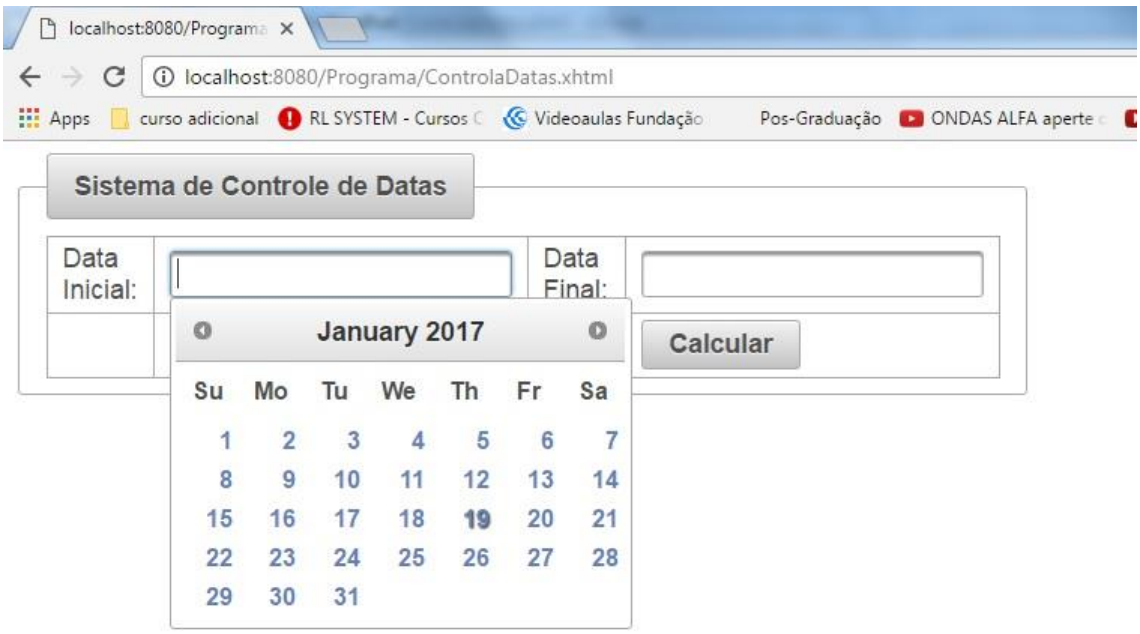

Figura 9 – Tela de Controle de Datas

Nesta tela o usuário irá encontrar uma mensagem informando o nome do sistema, “Sistema de Controle de Datas” e dois campos: data inicial e data final. No primeiro campo, quando o usuário clicar para inserir a data, automaticamente irá abrir um calendário onde será possível escolher a data desejada. Feito isto ao clicar no segundo campo: data final, ao usuário será apresentado novamente o calendário para que ele selecione outra data.

Após escolhidas as datas de inicio e fim, que representam as datas em que foi dada a ultima decisão processual e a data em que foi interposto o recurso pelo advogado da parte no Superior Tribunal de Justiça, o usuário deve clicar no botão “Calcular”. Então o sistema irá fazer um calculo de subtração da primeira e da ultima data. Caso o valor encontrado seja maior do que quinze dias uma mensagem na tela irá informar que o processo é intempestivo. Mas caso o prazo seja inferior a quinze dias a mensagem apresentada será positiva e o processo será tempestivo.

A tela de resposta do sistema é apresentada asseguir:

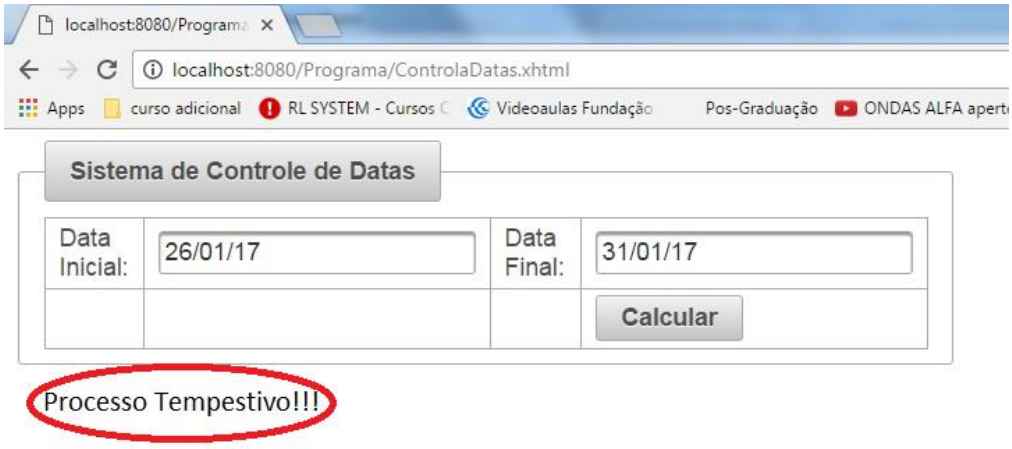

Figura 10 – Tela de Resposta

O código em JSF desta tela de controle de datas é apresentado logo abaixo e será explicado posteriormente.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:p="http://primefaces.org/ui">
<h:head>
</h:head>
<h:body>
  <h:form>
      <p:fieldset legend="Sistema de Controle de Datas"
style="width:340px">
     <p:panelGrid columns="4">
     <h:outputLabel value="Data Inicial: "></h:outputLabel>
     <p:calendar id="popup" value="#{CalcularDatas.data1}"/>
     <h:outputLabel value="Data Final: "/>
     <p:calendar id="popup2" value="#{CalcularDatas.data2}"/>
          <p:spacer></p:spacer>
          <p:spacer></p:spacer>
          <p:spacer></p:spacer>
     <p:commandButton value="Calcular"
actionListener="#{calcularDatas.calcular()} update="msg"/>
    <p:outputLabel id="msg" value="#{calcularDatas.resultado}"/>
  </p:panelGrid>
 </p:fieldset>
</h:form>
</h:body>
</html>
```

Figura 11 – Código Fonte Controle de Datas

Novamente iniciamos com a declaração normal de um arquivo xml e as definições de cabeçalho e corpo de um arquivo html. Dentro de nosso formulário incluímos um componente chamado: “<p:calendar” que é justamente um calendário popup que será apresentado ao usuário quando este clicar na caixa de inserção de texto.

Os valores escolhidos para data inicial e data final são enviados para no ManagedBean. Este é um componente responsável por armazenar as informações de nossa classe, no caso de nosso sistema a classe é “CalcularDatas”. Está classe possui dois atributos: data_inicio e data_fim que serão usadas para fazer o calculo da tempestividade. Esta classe ainda possui um método que é justamente o de calcular as datas que será chamado quando o usuário clicar no botão “calcular”.

A mensagem de confirmação positiva ou negativa será inserida em um componente chamado: “outputLabel” este componente mostra apenas um texto em nosso arquivo JSF. As mensagens são: “Processo Tempestivo” e “Processo Intempestivo”.

5. CONSIDERAÇÕES FINAIS

Neste projeto foram usadas algumas das tecnologias apresentadas no curso de pós-graduação oferecido pela faculdade integrada AVM. Dentre estas tecnologias podemos destacar o Java, JSF, JPA com Hibernate, Banco de dados Mysql dentre outras. Embarcando assim um grande nível de conhecimento em torno de desenvolvimento de sistemas usando a linguagem Java o qual era o propósito do curso.

Em nosso projeto em particular, o desafio era criar um sistema que facilitasse a rotina diária dos servidores do Superior Tribunal de Justiça, local em que trabalho, referente a pesquisa sobre pressupostos processuais e tempestividade de processos. A necessidade do trabalho surgiu pela observação do tempo gasto em analisar manualmente estes pressupostos para saber se um processo poderia ou não seguir seu curso normal dentro do tribunal.

Sendo assim foi construído um sistema que, permitiria o registro do usuário no sistema para que pudéssemos saber qual usuário estaria fazendo aquela consulta e posteriormente o sistema calcularia a diferença das datas baseadas em entradas fornecidas pelo usuário. Caso a data de inicio e a data de fim não ultrapassasse o valor de 15 dias o processo seria tempestivo e o sistema apresentaria uma mensagem positiva permitindo ao servidor dar prosseguimento normal ao processo.

Ao realizarmos a analise do sistema foi optado por escolher a linguagem JSF pois esta proporciona um ambiente muito produtivo em relação às telas de comunicação com o usuário. Com a implementação PrimeFaces, este framework prove um conjunto de elementos já prontos para que possamos usar em nossas telas do sistema.

A finalidade do projeto foi alcançada já que o tempo gasto na analise de pressupostos processuais diminui consideravelmente e com isso podemos dar agilidade

na rotina da sessão de analise de pressupostos. O sistema é fácil de utilizar e cumpre seu proposito de analisar as datas e informar ao usuário o resultado.

REFERÊNCIAS BIBLIOGRÁFIAS

PRESSMAN, Roger S. Engenharia de Software. São Paulo. Ed. Markon Books, 1995

SOMERVILLE, Engenharia de software. 6° ed. São Paulo. Ed Addison-Wesley, 2003

Beck, K. Programação Extrema Explicada. Ed Bookman, 1999.

Agile Manifesto (2004) Disponível em http://agilemanifesto.org/,
acessado em 22 de Dezembro de 2016.

TANENBAUM, A. S. Operating Systems: Design and Implementation. 1987.

SILBERSCHATZ, A.; PETERSON, J. L.; GAVIN, P. Conceitos de Sistemas Operacionais. Ed. Addison-Wesley, 1992.

DEITEL, H. M.; DEITEL, P. J.; CHOFFNES, D. R. Sistemas operacionais. São Paulo. Ed. Pearson, 2005.

DENNIS, Alan; WIXOM, Barbara. Análise e projetos de sistemas. 2005